Long Naïve Bayes question

Question

You have a set of book reviews. In the left column of the following table, each of "a", "b", "c" represents a different word. In the right column, "1" means the review is positive and "0" otherwise.

Review Text	Positive
aaab	1
cbaba	1
aba	1
cb	1
bccb	0
acc	0
abbc	0

If you want to use the review text to predict its tone (positive or not) with Naive Bayes Classification:

i) What assumption does Naive Bayes make on the data?

Assuming the naive assumption is met, and you have an unlabeled piece of review text to classify: "bc". When answering the following questions, please write down the steps.

ii) Compute the prior p(tone=1), p(tone=0), p(b), and p(c). DO **NOT** use any smoothing method. Present the answer as fractions.

iii) Compute the posterior p(tone=1|bc) and p(tone=0|bc). Present the answer as decimals with 3 digit precision. What is the tone of this review?

Answer

- Feature attributes are conditionally independent conditioned on the label; (0.5 pts)

- priors:
  P(tone=1) = 4/7;
  P(tone=0) = 3/7
  P(b) = 9/25
  P(c) = 7/25
  (1.5 pts)

- Posteriors:

$$\mathrm{P(bc)} = P(tone=1)\,P(b|tone=1)\,P(c|tone=1) + \mathrm{P}(tone=0)\,P(b|tone=0)\,P(c|tone=0)$$
$$= \frac{4}{7}\times\frac{5}{14}\times\frac{2}{14}+\frac{3}{7}\times\frac{4}{11}\times\frac{5}{11}$$

$P(tone=1 \mid bc) = \frac{P(tone=1)\,P(b|tone=1)\,P(c|tone=1)}{P(bc)} = \frac{\frac{4}{7}\times\frac{5}{14}\times\frac{2}{14}}{\frac{4}{7}\times\frac{5}{14}\times\frac{2}{14}+\frac{3}{7}\times\frac{4}{11}\times\frac{5}{11}} = 0.292$ (0.4 pts; OK if differ slightly in the last digits)

$P(tone=0 \mid bc) = \frac{P(tone=0)\,P(b|tone=0)\,P(c|tone=0)}{\mathrm{P(bc)}} = \frac{\frac{3}{7}\times\frac{4}{11}\times\frac{5}{11}}{\frac{4}{7}\times\frac{5}{14}\times\frac{2}{14}+\frac{3}{7}\times\frac{4}{11}\times\frac{5}{11}} = 0.708$ (0.4 pts; OK if differ slightly in the last digits)

negative because $P(tone=0 \mid bc) > P(tone=1 \mid bc)$ (0.2 pts)

===

Question

You have a set of book reviews. In the left column of the following table, each of "a", "b", "c" represents a different word. In the right column, "1" means the review is positive and "0" otherwise.

Review Text	Positive
baa	0
bcaba	0
abaa	0
bc	0
babc	1
cac	1
bbcc	1

If you want to use the review text to predict its tone (positive or not) with Naive Bayes Classification:

i) What assumption does Naive Bayes make on the data?

Assuming the naive assumption is met, and you have an unlabeled piece of review text to classify: "cb". When answering the following questions, please write down the steps. If the answers are fractions, you can but don't need to convert them to decimals.

ii) Compute the prior p(tone=1), p(tone=0), p(b), and p(c). DO **NOT** use any smoothing method. Present the answer as fractions.

iii) Compute the posterior p(tone=1|cb) and p(tone=0|cb). Present the answer as decimals with 3 digit precision. What is the tone of this review?

Answer:

- Feature attributes are conditionally independent conditioned on the label; (0.5 pts)

- priors:

  P(tone=1) = 3/7;
  P(tone=0) = 4/7
  P(b) = 9/25
  P(c) = 7/25
  (1.5 pts)

- Posteriors:

$$\mathrm{P(cb)} = P(tone = 1)\, P(b|tone = 1)\, P(c|tone = 1) + P(tone = 0)\, P(b|tone = 0)\, P(c|tone = 0) = \frac{3}{7} \times \frac{4}{11} \times \frac{5}{11} + \frac{4}{7} \times \frac{5}{14} \times \frac{2}{14}$$

$P(tone = 1 \mid cb) = \frac{P(tone=1)\, P(b|tone = 1)\, P(c|tone = 1)}{P(\mathrm{cb})} = \frac{\frac{3}{7}\times\frac{4}{11}\times\frac{5}{11}}{\frac{3}{7}\times\frac{4}{11}\times\frac{5}{11}+\frac{4}{7}\times\frac{5}{14}\times\frac{2}{14}} = 0.708$ (0.4 pts; OK if differ slightly in the last digits)

$P(tone = 0 \mid cb) = \frac{P(tone=0)\, P(b|tone = 0)\, P(c|tone = 0)}{P(\mathrm{cb})} = \frac{\frac{4}{7}\times\frac{5}{14}\times\frac{2}{14}}{\frac{3}{7}\times\frac{4}{11}\times\frac{5}{11}+\frac{4}{7}\times\frac{5}{14}\times\frac{2}{14}} = 0.292$ (0.4 pts; OK if differ slightly in the last digits)

positive because $P(tone = 1 \mid bc)$> $P(tone = 0 \mid bc)$ (0.2 pts)

Long CNN:

Q:
Your input is a 58 * 91 RGB image, and you use a convolutional layer with 100 filters that are each 6 * 5 and with one bias parameter for each filter, using a stride of (4,6) and a total padding of 4.

1. What is the output shape?

2. What is the number of parameters (including the bias parameters)?

3. Suppose we further use a 3 * 4 max-pooling layer (with a stride of (3,4) and no padding) after the above convolutional layer. What is the output shape of the max-pooling layer?

Answer: (1) 15 * 16 * 100
(2) 9100
(3) 5 * 4 * 100

Alternative Q:
Your input is a 131 * 63 RGB image, and you use a convolutional layer with 100 filters that are each 7 * 4 and with one bias parameter for each filter, using a stride of (4,7) and a total padding of 4.

1. What is the output shape?

2. What is the number of parameters (including the bias parameters)?

3. Suppose we further use a 3 * 2 max-pooling layer (with a stride of (3,2) and no padding) after the above convolutional layer. What is the output shape of the max-pooling layer?

Answer: (1) 33 * 10 * 100
(2) 8500
(3) 11 * 5 * 100

Long RL question:

Version 1

Q1 (1.5pts)

V*(S1) = R(S1)=10, V*(S2) = R(S2)=20. (0.5pts)

Expected utility for different actions:

Action A1: V = 5 + 0.8(0.1 * 10 + 0.9 * 20) = 20.2

Action A2: V = 5 + 0.8(0.9 * 10 + 0.1 * 20) = 13.8

Action A3: V = 5 + 0.8(0.5 * 10 + 0.5 * 20) = 17

Therefore, the optimal policy from state S0 is to take Action A1. (expected utility + optimal action: 1pts)

Q2 (0.5pts)

The notable difference between Q-learning and SARSA is that when updating Q(s,a): Q-learning uses the expected cumulative discounted reward for taking an **optimal** action from state s' whereas SARSA uses the expected cumulative discounted reward for the **actual** action it took from state s'. Also, SARSA also updates Q(s', a') in each iteration.

Q3(0.5pts)

The Epsilon-greedy algorithm enables a balance between exploration and exploitation, so that both SARSA and Q-learning could bootstrap with non-initial value (stuck).

Q4 (0.5pts)

The max is 8: each state leads to an update to a distinct entry.

The min is 1: all observations lead to updates to the same (s,a) entry.

Version 2

Q1(1.5pts)

V*(S1) = R(S1)=10, V*(S2) = R(S2)=25. (0.5pts)

Expected utility for different actions:

Action A1: 10 + 0.9(0.8 * 10 + 0.2 * 25) = 21.7

Action A2: V = 10 + 0.9(0.2 * 10 + 0.8 * 25) = 29.8

Action A3: V = 10 + 0.9(0.5 * 10 + 0.5 * 25) = 25.75

Therefore, the optimal policy from state S0 is to take Action A2. (expected utility + optimal action: 1pts)

Q2 (0.5pts)

The notable difference between Q-learning and SARSA is that when updating Q(s,a): Q-learning uses the expected cumulative discounted reward for taking an **optimal** action from state s' whereas SARSA uses the expected cumulative discounted reward for the **actual** action it took from state s'. Also, SARSA also updates Q(s', a') in each iteration.

Q3 (0.5pts)

The Epsilon-greedy algorithm enables a balance between exploration and exploitation, so that both SARSA and Q-learning could bootstrap with non-initial value (stuck).

Q4 (0.5pts)

The max is 8: each state leads to an update to a distinct entry.

The min is 1: all observations lead to updates to the same (s,a) entry.

Long Search question:

Version 1: Double star graph:

A BFS will start at 0 and then consider every node one edge away, which are 1,2,...,n. By the tie-breaking rule, it will follow in that order. Then, after node n is visited, every node in the second layer is visited, which are n+1, n+2, ..., 2n.

A DFS will start at 0 then visit 1 then 2 at which point it has to back up back to 0. This process is repeated: visiting i then i+n from 0 and then continuing on with i+1 till all nodes are visited, eventually ending with n and 2n.

Version 2: wheel graph:

A BFS will start at 0 and then consider every node one edge away, which are 1, 2…, n. By the tie-breaking rule, it will follow in that order.

A DFS will start at 0 then visit 1 at which point it follows the outer "cycle" 1,2,3…,n.